UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/713,409	11/14/2003	Adam H. Leventhal	03226.347001; SUN040249	7012

32615 7590 02/08/2007

OSHA LIANG L.L.P./SUN
1221 MCKINNEY, SUITE 2800
HOUSTON, TX 77010

EXAMINER	
VO, TED T	
ART UNIT	**PAPER NUMBER**
2191	

SHORTENED STATUTORY PERIOD OF RESPONSE	MAIL DATE	DELIVERY MODE
3 MONTHS	02/08/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Office Action Summary	Application No.	Applicant(s)	
	10/713,409	LEVENTHAL ET AL.	
	Examiner	Art Unit	
	Ted T. Vo	2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *14 November 2003*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-16* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-16* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *14 November 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *3/29/04*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to the communication filed on 11/14/2003.

Claims 1-16 are pending in the application.

Double Patenting

2. The nonstatutory double patenting rejection is based on a judicially created doctrine grounded in public policy (a policy reflected in the statute) so as to prevent the unjustified or improper timewise extension of the "right to exclude" granted by a patent and to prevent possible harassment by multiple assignees. See *In re Goodman*, 11 F.3d 1046, 29 USPQ2d 2010 (Fed. Cir. 1993); *In re Longi*, 759 F.2d 887, 225 USPQ 645 (Fed. Cir. 1985); *In re Van Omum*, 686 F.2d 937, 214 USPQ 761 (CCPA 1982); *In re Vogel*, 422 F.2d 438, 164 USPQ 619 (CCPA 1970); and, *In re Thorington*, 418 F.2d 528, 163 USPQ 644 (CCPA 1969).

A timely filed terminal disclaimer in compliance with 37 CFR 1.321(c) may be used to overcome an actual or provisional rejection based on a nonstatutory double patenting ground provided the conflicting application or patent is shown to be commonly owned with this application. See 37 CFR 1.130(b).

Effective January 1, 1994, a registered attorney or agent of record may sign a terminal disclaimer. A terminal disclaimer signed by the assignee must fully comply with 37 CFR 3.73(b).

3. Claims 1-16 are provisionally rejected on the ground of nonstatutory double patenting over claims 1-20 of copending Application No. **10/713,411, available in the** Attorney Docket Number: **03226.346001; SUN040247.** This is a provisional double patenting rejection since the conflicting claims have not yet been patented. The subject matter claimed in the instant application is fully disclosed in the referenced copending application and would be covered by any patent granted on that copending application since the referenced copending application and the instant application are claiming common subject matter, as follows: Claims 1-20 of the US patent application serial No. **10/713,411** has all claimed subject matters presenting in the instant claims 1-16. Furthermore, there is no apparent reason why applicant would be prevented from presenting claims corresponding to those of the instant application in the other copending application. See *In re Schneller*, 397 F.2d 350, 158 USPQ 210 (CCPA 1968). See also MPEP § 804.

Claim Rejections - 35 USC § 101

4. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

5. The claims 10-16 are rejected under 35 U.S.C 101 because the claimed invention is directed to non-statutory subject matter.

As per claims 10-16, the claims recite a system for tracing an instrumented program, where the system fails to be tangibly embodied. No limitations in the claims show the system as being a hardware-embodied system. Thus, the system is software/program per se. This type of claiming fails to be statutory claim.

Claim Rejections - 35 USC § 102

6. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

7. Claims 1-16 are rejected under 35 U.S.C. 102(b) as being anticipated by Uhlig et al., "Trace-Driven Memory Simulation: A Survey", ACM, 6-1997.

Given the broadest reasonable interpretation of followed claims in light of the specification.

As per Claim 1: Uhlig discloses,

A method for tracing an instrumented program using a thread (emulation method: Figure 3, p.139), ***comprising:***

transferring control of the instrumented program to a trap handler ('Handler': See Figure 3, the middle section: the control is transferred to a 'Handler") ***to obtain an original instruction*** (e.g. ld r2, 4(r1), stored in Registers) *associated* ***with a probe*** (See a program in Translation Table/Cache, having the original instruction and some trace instruction 'probe' such as trace(t2));

loading the original instruction into a scratch space (e.g., see "ld r2, 4(r1)" in the Translation Table/Cache - Translation Table/Cache: *a scratch space*);

setting a program counter to point to the scratch space (See instruction in Handler, e.g. trace(i ++));

setting a next program counter to point to a next instruction (See instruction in Handler, e.g. trace(i++)); *and*

executing the original instruction in the scratch space using the thread (the execution is using Translation table/cache), ***wherein***

executing the original instruction results in placing the instrumented program in a state equivalent to natively executing the original instruction (the emulation executes of the program in the Translation table/cache, which is equivalent to the execution of this program in the hardware registers in the processor's memory, for the trace purpose).

As per Claim 2: Uhlig discloses, *The method of claim 1, further comprising:*

determining whether the original instruction is a control-flow instruction (referred to branch instruction, e.g., see p. 139, "Spa"); *and*

emulating a location dependent instruction in a kernel if the original instruction is a control-flow instruction, wherein semantics of the location dependent instruction depend on a location of the original instruction within the instrumented program (i.e., a Branch instruction is used in the place of, "ld r2, 4(r1)", and see discussion of kernel references, p. 138, left col., within sec 4.2).

As per Claim 3: Uhlig discloses, *The method of claim 2, further comprising:*

updating the program counter and the next program counter using a result from emulating the original instruction in the kernel if the original instruction is control-flow instruction (See Figure 3, the "Handler" is has the ability to update machine states (see p. 139, right col. with this mechanism, program counter and next program counter when it uses any instructions. For instance, in the particular case of "Spa" or of kernel, the address of the used instruction will be handled by the handler).

As per Claim 4: Uhlig discloses, *The method of claim 1, further comprising:*

triggering the probe in the instrumented program (e.g. the emulation of the program shown in the Translation Table/Cache).

As per Claim 5: Uhlig discloses, *The method of claim 1, wherein the probe corresponds to a trap instruction* (The reference discusses trap-driven simulation).

As per claim 6: Uhlig discloses,

The method of claim 1, wherein obtaining the original instruction comprises:

searching a look-up table using the program counter (See Figure 3, "Lookup"), *wherein the look-up table contains the original instruction associated with the probe and an address associated with the original instruction* (see, 'Lookup' points to 'opcode Ra Rb Offset'/Dispatch).

As per claim 7: Uhlig discloses, *The method of claim 1, wherein the scratch space is allocated on a per-thread basis* (See portion of the program in the Translation Table/Cache).

As per claim 8: Uhlig discloses, *The method of claim 1, wherein the instrumented program is executed on multi-thread architecture* (The reference supports multithreaded applications, see p. 141, left col.).

As per claim 9: Uhlig discloses, *The method of claim 1, wherein loading the original instruction comprises using a block copy instruction* (The reference supports sequential blocks of code, see p. 140, right col. line 6).

As per claim 10: Uhlig discloses, ***A system for tracing an instrumented program, comprising:***

a program counter configured to store a current address corresponding to a current instruction in the instrumented program (See Figure 3, p. 139, program counter used in the Handler);

a next program counter configured to store a next address corresponding to a next instruction in the instrumented program (See Figure 3, program counter used in the Handler);

a scratch space arranged to store an original instruction (Figure 3, Translation Table /Cache);

a thread configured to execute the instrumented program and the original instruction (the emulation)***; and***

a trap handler configured to halt execution of the thread when a trap instruction is encountered (trace-driven simulation), ***to obtain the corresponding original instruction from a look-up table using an address of the trap instruction, and to set the program counter to the scratch space.***

See rationale addressed in the rejection of Claim 1.

As per claim 11: Uhlig discloses,. *The system of claim 10, further comprising:*

a buffer for storing the data (Figure 3, p. 139, trace buffer).

As per claim 12: Uhlig discloses, *The system of claim 10, further comprising:*

a kernel configured to emulate a location dependent instruction if the original instruction is a control-flow instruction (lower part of Figure 3),

wherein semantics of the location dependent instruction depend on a location of the original instruction within the instrumented program (See the content of instructions in the lower part of figure 3).

As per claim 13: MSF discloses, *The system of claim 10, further comprising:*

a look-up table configured to store the address and the original instruction (See Figure 3, "lookup' is pointed to the location of the emulated program in the Translation Table/Cache).

As per claim 14: MSF discloses, *The system of claim 10, wherein the scratch space is allocated on a per-thread basis* (Refer to the instruction in the Handler).

As per claim 15: MSF discloses, *The system of claim 10, wherein the instrumented program is executed on multi- thread architecture* (The reference support the multi-thread architecture).

As per claim 16: MSF discloses, *The system of claim 10, wherein the trap handler is configured to transfer control to the thread prior to the thread executing the original instruction* (Refer the original instruction as ld r2, 4(r1), see the flow process in Figure 3, that is done before the emulation).

8. Claims 1-16 are rejected under 35 U.S.C. 102(b) as being anticipated by Tamches, "Fine-Grained Dynamic Instrumentation of Commodity Operating System Kernels", University of Wisconsin, 2001.

Given the broadest reasonable interpretation of followed claims in light of the specification.

As per Claim 1: Tamches discloses,

A method for tracing an instrumented program using a thread comprising:

transferring control of the instrumented program (e.g., Figure 4.1, p. 49, particularly, text in p. 52, first full paragraph) ***to a trap handler*** (see p. 13, sec. 2.1.4, third paragraph; see p.14, sec. 2.1.5. See Figure 4.1, p. 49) ***to obtain an original instruction associated with a probe*** ('Code Patch' at Instrumentation point);

loading the original instruction into a scratch space (See Figure 3.1, p. 27, Code Patch for privileged user programs and kernel modules is attached into a kernel address space);

setting a program counter to point to the scratch space (Return to Figure 4.1, Kernel code is refer to a kernel module, and "Code Patch" is now located in an available Kernel addressed space, a program counter points code patch. For example, see p. 52, second full paragraph);

setting a next program counter to point to a next instruction (A program counter is point to next instruction, according to the execution)*; and*

executing the original instruction in the scratch space using the thread (the patch code is executed at boot time or run-time of the kernel), ***wherein***

executing the original instruction results in placing the instrumented program in a state equivalent to natively executing the original instruction (See Figure 4.1, the statement in the description of Code Patch: Overwritten instruction or equivalent sequence).

As per Claim 2: Tamches discloses, *The method of claim 1, further comprising:*

determining whether the original instruction is a control-flow instruction (referred to branch instruction, in figure 4.1); *and*

emulating a location dependent instruction in a kernel if the original instruction is a control-flow instruction, wherein semantics of the location dependent instruction depend on a location of the original

instruction within the instrumented program (i.e., p. 32-33, discussions in all bold bullet, and see p. 52, second full paragraph).

As per Claim 3: Tamches discloses, *The method of claim 2, further comprising:*

updating the program counter and the next program counter using a result from emulating the original instruction in the kernel if the original instruction is control-flow instruction (i.e., p. 32-33, discussions in all bold bullet, and see p. 52, second full paragraph).

As per Claim 4: Tamches discloses, *The method of claim 1, further comprising:*

triggering the probe in the instrumented program (e.g. encountering the instrumentation point – See further IBM Dprobes, p. 15).

As per Claim 5: Tamches discloses, *The method of claim 1, wherein the probe corresponds to a trap instruction* (See discussing on p. 13-14).

As per claim 6: Tamches discloses, *The method of claim 1, wherein obtaining the original instruction comprises:*

searching a look-up table using the program counter (Note IBM Probes is part of this Tamches' disclosure, *a look-up table,* is instrumentation site handled by trap handler. Furthermore, see p. 66, sec. 4.6.1, discussing a trap handler, it looks up the trap address in a hash table), *wherein the look-up table contains the original instruction associated with the probe and an address associated with the original instruction* (i.e. Code Patch such as seen in figure 4.1).

As per claim 7: Tamches discloses, *The method of claim 1, wherein the scratch space* (Kernel space or scratch register which is available for storing patch code) *is allocated on a per-thread basis* (that is the allocation of each instrumentation point with respect to a code patch used by the frame work of Figure 3.1).

As per claim 8: Tamches discloses, *The method of claim 1, wherein the instrumented program is executed on multi-thread architecture* (The reference supports multithreaded applications, see p. 55, sec. 4.3).

As per claim 9: Tamches discloses, *The method of claim 1, wherein loading the original instruction comprises using a block copy instruction* (The reference supports blocks of code, see p. 42).

As per claim 10: Tamches discloses, ***A system for tracing an instrumented program, comprising:***

a program counter configured to store a current address corresponding to a current instruction in the instrumented program (See Figure 3.1, p. 27, see Figure 4.1, p. 49, and see p. 52, refer to 'program counter', this program counter is used in control transfer instruction of code patch);

a next program counter configured to store a next address corresponding to a next instruction in the instrumented program (see Figure 4.1, p. 49)***;***

a scratch space arranged to store an original instruction (Figure 3.1, e.g. Kernel address space);

a thread configured to execute the instrumented program and the original instruction (the execution of patch code at the kernel address space)***; and***

a trap handler configured to halt execution of the thread when a trap instruction is encountered (trace-driven simulation), ***to obtain the corresponding original instruction from a look-up table using an address of the trap instruction, and to set the program counter to the scratch space.***

See rationale addressed in the rejection of Claim 1.

As per claim 11: Tamches discloses, *The system of claim 10, further comprising:*

a buffer for storing the data (kernel memory or data cache, or log information in a computer).

As per claim 12: Tamches discloses, *The system of claim 10, further comprising:*

a kernel configured to emulate a location dependent instruction if the original instruction is a control-flow instruction (See Figure 4.1, and p, 52, second full paragraph),

wherein semantics of the location dependent instruction depend on a location of the original instruction within the instrumented program (See p, 52, second full paragraph).

As per claim 13: Tamches discloses, *The system of claim 10, further comprising:*

a look-up table configured to store the address and the original instruction (Note IBM Probes is part of this Tamches' disclosure, where *look-up table,* is instrumentation site handled by trap handler, i.e. Code Patch such as seen in figure 4.1. Furthermore, see p. 66, sec. 4.6.1, discussing a trap handler, it looks up the trap address in a hash table).

As per claim 14: MSF discloses, *The system of claim 10, wherein the scratch space is allocated on a per-thread basis* (Refer to the allocation of instrumentation points).

As per claim 15: MSF discloses, *The system of claim 10, wherein the instrumented program is executed on multi-thread architecture* (The reference support the multi-thread architecture).

As per claim 16: MSF discloses, *The system of claim 10, wherein the trap handler is configured to transfer control to the thread prior to the thread executing the original instruction* (See sec. 4.6.1, p. 66).

Conclusion

9. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ted T. Vo whose telephone number is (571) 272-3706. The examiner can normally be reached on 8:00AM to 4:30PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Y. Zhen can be reached on (571) 272-3708.

The facsimile number for the organization where this application or proceeding is assigned is the Central Facsimile number **571-273-8300**.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

TTV
February 02, 2007

TED VO
PRIMARY EXAMINER
TECHNOLOGY CENTER 2100